CS-470 Final Reflection

Joshua Kovacevich, SNHU

2022/12/18

https://www.youtube.com/watch?v=hJ8z05q_ngg&ab_channel=JoshKovacevich

I think this course has helped me gain significant insight into the workings of the cloud and AWS as a whole. This is a technology I had very limited exposure to/experience with prior to taking this course. I think this knowledge will be generally easy to build upon when working with cloud applications in the future or setting up cloud related storage systems. To be honest while I do not think I have mastered the cloud by any means (I think very few people have) I do think I have gained a solid foundation in web development through this and the previous related Full Stack development course. As a software developer I have some strengths in various technologies, though I have been primarily focused on Java and C++ in recent months, as Java is what I work in currently and C++ will always be a favorite language for me. One of the major strengths I have now is that I can pick up new or spin-off technologies at a relatively fast pace as I have a fair amount of experience learning software development, frameworks, and languages at this point, though a good share of this came from nearly 2 decades of being a hobbyist before this course. While I am currently in a role as a Software Developer I, I think that I am getting closer every day to being a Software Developer II, and likely will be ready to step into that role in the next year, which will involve me being able to lead teams through projects and handle mentoring and training/teaching newer developers and interns better coding practices.

I think one of the biggest benefits to cloud microservices and serverless hosting is that you can access data from anywhere with a high likelihood of availability. Additionally, cloud resources can be leveraged to store and compile from numerous decentralized data sources which can help larger management and monitoring across distributed environments. Serverless hosting allows you to utilize resources like docker compose to automate scaling and error handling, which once properly set up can help you spend less time on application management and spend more time on development and improvement. Personally, I think predicting cost through cloud hosting is probably one of the most difficult things related to the cloud, that being said almost all major cloud providers allow you to walk through calculators to estimate costs and generally plan your application infrastructure in the most economically feasible manner for your specific needs. Personally, I think containers are more predictable for estimating cost as you know the resource utilization of each container, and the capabilities. When it comes to completely serverless with lambdas there is a bit of an unknown in real costs, and there is the often-undiscussed factor related to the speed of response in web applications. While Lambdas are generally acceptable (in speed and resource terms) for many applications, if your application is dependent on them, it is likely that you will lose customers due to the lead times and overhead costs associated with spinning up these virtual functions.

In terms of pro and cons I think that the application of the cloud really determines which solution is better. I think that if you are hosting a single page application on the cloud you are probably fine but hosting a gaming server in the cloud is likely not reasonable with a highly tuned design which may cost significantly more to design and develop than a traditional gaming service. The pros and cons that stay the same regardless of application are usually scalability and iteration. The cloud allows for constant iteration without the same limitations a normal system may have, and also allows infinite (or near infinite) scalability with a good initial design. Elasticity and pay-for-service is great in this regard as you are not generally paying for services or resources you do not need, and when you do need them, they can be scaled automatically. This is great for situations where x amount of traffic results in y amount of revenue, and not so good when y amount of revenue drastically varies with only some consideration for x amount of traffic.